Please amend the present application as follows:

Specification

The following is a marked-up version of the specification with the language that is underlined ("___") being added and the language that contains strikethrough ("—") being deleted:

Page 23, line 22 through page 24, line 15.

If the fragment is not to be replaced, for instance is free of bugs, flow continues to block 518 described below. If, on the other hand, the fragment includes code that is to be replaced, flow continues to block 516 at which the application code is replaced with replacement code, *i.e.*, one or more instructions that provide the desired function. By way of example, this replacement code is stored within computing system memory 304 beyond the DELI 100 so that it cannot be accidentally deleted during DELI operation. The correct replacement code can be fetched from the storage location by the DELI 100 with reference to an appropriate identifier (*e.g.*, tag) contained in the descriptor of the patch. The replacement of the original code also entails changing all references to that code such that these references will in the future direct execution to the replacement code (instructions). In most cases, the DELI 100 will intercept all branches to entry points of the replaced code (function) and replace them with branches to the new code. The DELI 100 can easily scan for direct branches to the function entry point by inspecting the target address of each branch instruction. As for indirect branches, the DELI 100, at ~~At~~ the time of uploading the patch, ~~the DELI 100~~ can inspect this table and remove all targets that fall into the old address range. When the DELI 100 attempts to branch indirectly to an address that is

not in this table, it returns to its emulator, and then emulates the call to the patched routine, executing the new code.